Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 16

По дисциплине «Основы алгоритмизации и программирования»

На тему «Функции пользователя»

Выполнила:

Студент(ка) 1 курса 7 группы

Подшиваленко Диана Игоревна

Преподаватель: асс. Андронова М.В.

2023, Минск

|  |  |
| --- | --- |
| **11** | 1. Дан массив размерностью **N** x **M**. Найти максимальный элемент и его позицию. Если максимальных элементов несколько, то вывести их все. Найти сумму элементов ниже главной диагонали.  2. Преобразовать строку: после каждой буквы **c** добавить символ **\*.** |

#include <iostream>

#include <iomanip>

#include <Windows.h>

#include <stdio.h>

#include <conio.h>

using namespace std;

int max\_ind(int \*\*a, int n, int m) { // функция для нахождения максимального элемента массива

int x = a[0][0];

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

if (a[i][j] > x) {

x = a[i][j];

}

}

}

return x;

}

int\*\* poisk(int\*\* a, int n, int m, int max, int &kol) { // функция для нахождения количества максимальных элементов и их позиций

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

if (a[i][j] == max) {

kol += 1;

}

}

}

int k = 0;

int \*\*x = new int\*[2];

for (int j = 0; j < 2; j++) {

x[j] = new int[kol];

}

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

if (a[i][j] == max) {

x[0][k] = i;

x[1][k] = j;

k += 1;

}

}

}

return x;

}

void vivod\_pos(int\*\* pos, int n) { // функция для вывода позиций максимального элемента

cout << "Позиции максимальных элементов в массиве" << endl;

for (int i = 0; i < n; i++) {

cout << "(" << pos[0][i] << ", " << pos[1][i] << ")" << " ";

}

cout << endl;

}

int sumn(int\*\* a, int n, int m) { // функция для вычисления суммы элементов, которые стоят ниже главной диагонали

int s = 0;

for (int i = 1; i < n; i++) {

for (int j = 0; j < i; j++) {

s += a[i][j];

}

}

return s;

}

char \*new\_str(char\* a, int &n, int pos) { // функция для добавления символа '\*' в строку после символа 'с'

n++;

char \*m = new char[n + 1];

for (int i = 0; i < n; i++) {

if (i != pos) {

m[i] = a[i];

}

else {

m[i] = '\*';

break;

}

}

for (int i = pos + 1; i < n; i++) {

m[i] = a[i - 1];

}

return m;

}

int main() { // функция main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do{

cout << "Выберите задачу, которую будем решать" << endl;

cout << "1 - Нахождение максимального элемента в матрице, количества его вхождений в матрицу и суммы элементов, которые расположены ниже главной диагонали " << endl;

cout << "2 - Преобразование строки: после каждой буквы с добавить символ \*" << endl;

cout << "3 - выход" << endl;

cin >> choice; // считываем выбор пользователя

switch (choice) { // создаём оператор выбора между задачами и опцию выхода

case 1: {

int n, m;

cout << "Введите размер матрицы(количество строк и столбцов) ";

cin >> n >> m; // считываем размер матрицы

int\*\* a = new int\*[n];

for (int i = 0; i < n; i++) { // считываем матрицу

a[i] = new int[m];

for (int j = 0; j < m; j++) {

cin >> a[i][j];

}

}

int max = max\_ind(a, n, m); // вызываем функцию для нахождения максимального элемента

cout << "Максимальный элемент массива = " << max << endl;

int kol = 0;

int \*\*pos = poisk(a, n, m, max, kol); // вызываем функцию поиска количества и позиций максимального элемента

vivod\_pos(pos, kol); // вызываем функцию для вывода позиций максимального элемента

int sum = sumn(a, n, m); // вызываем функцию для нахождения суммы элементов, которые стоят ниже главной диагонали

cout << "Сумма элементов, которые стоят ниже главной диагонали = " << sum << endl;

delete[] a;

break;

}

case 2: {

char \*a; // создаём указатель на массив символов

cout << "Введите размер строки ";

int n;

cin >> n; // считываем размер строки

a = new char[n + 1];

cout << "Введите строку" << endl;

cin >> a; // считываем строку

for (int i = 0; i < n; i++) { // ищем символ 'с'

if (a[i] == 'c') { // если нашли, то

a = new\_str(a, n, i + 1); //вызываем функцию для вставки символа '\*'

}

}

for (int i = 0; i < n; i++) { // выводим строку

cout << a[i];

}

cout << endl;

delete[] a;

break;

}

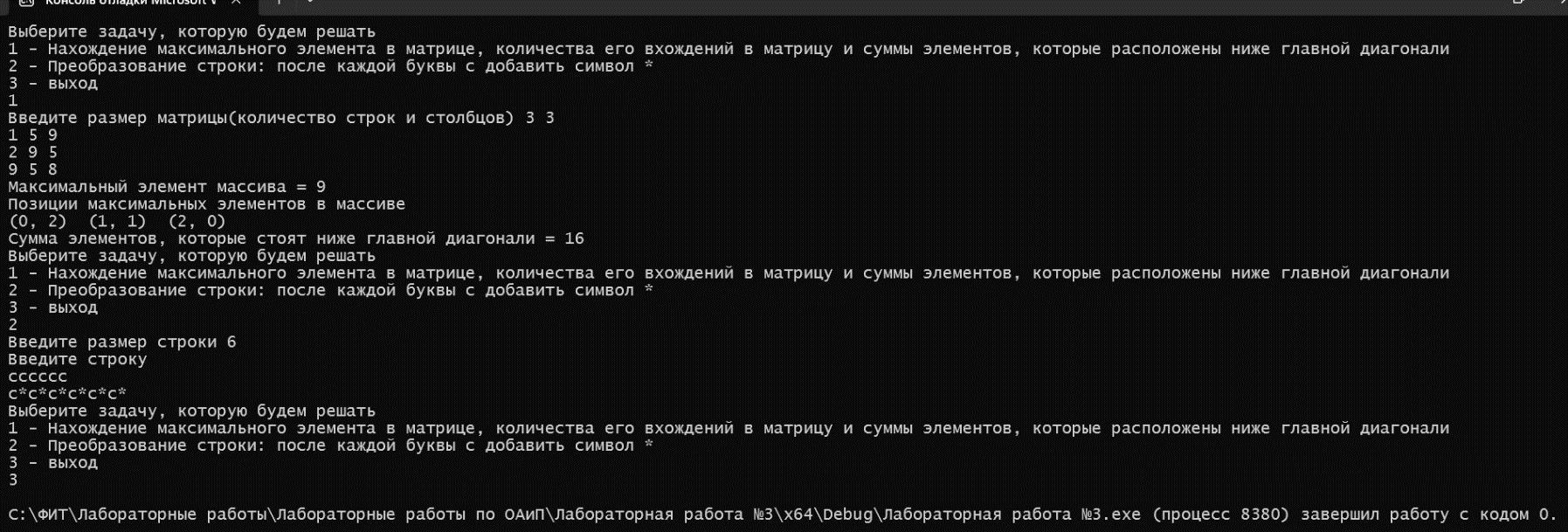
case 3:{ // если пользователь выбрал выход, то прерываем программу

break;

}

}

} while (choice != 3);

}

|  |  |
| --- | --- |
| **2** | 1. Ввести вещественные числа в массив **А**[**n**, **m**] (**n**, **m** > 0) и ко всем числам каждой строки прибавить единицу. Если первый элемент строки нулевой, ничего не делать.  **2.** Дана строка. Все русские буквы привести к верхнему регистру, латинские − заменить символом «**?**». Вывести результат на экран. |

#include <iostream>

#include <iomanip>

#include <Windows.h>

#include <stdio.h>

#include <conio.h>

using namespace std;

double\*\* zam(double\*\* a, int n, int m) { // функция для добавления 1 к элементам матрицы

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

if (a[i][j] == 0 and j == 0) {

continue;

}

else {

a[i][j] += 1;

}

}

}

return a;

}

void vivod(double\*\* a, int n, int m) { // функция вывода матрицы

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

cout << a[i][j] << " ";

}

cout << endl;

}

}

char \*zamena(char\* a, int n) { // функция замены символов в строке

for (int i = 0; i < n; i++) {

if (a[i] >= 'а' and a[i] <= 'я') {

a[i] = a[i] - 'а' + 'А';

}

else if ((a[i] >= 'A' and a[i] <= 'Z') or (a[i] >= 'a' and a[i] <= 'z')) {

a[i] = '?';

}

}

return a;

}

int main() { // функция main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do{

cout << "Выберите задачу, которую будем решать" << endl;

cout << "1 - Добавление 1 к каждому элементу массива, кроме 0, стоящих на 1 позиции " << endl;

cout << "2 - Преобразование строки: Все русские буквы привести к верхнему регистру, латинские - заменить символом «?»" << endl;

cout << "3 - выход" << endl;

cin >> choice; // считываем выбор пользователя

switch (choice) { // создаём оператор выбора между задачами и опцию выхода

case 1: {

int n, m;

cout << "Введите размер матрицы(количество строк и столбцов) ";

cin >> n >> m; // считываем размер матрицы

double \*\*a = new double\*[n];

for (int i = 0; i < n; i++) { // считываем матрицу

a[i] = new double[m];

for (int j = 0; j < m; j++) {

cin >> a[i][j];

}

}

a = zam(a, n, m); // выполняем функцию zam

vivod(a, n, m); // выводим матрицу с помощью функции vivod

delete[] a; // освобождаем память

break;

}

case 2: {

char \*a; // создаём указатель на массив символов

cout << "Введите размер строки ";

int n;

cin >> n; // считываем размер строки

a = new char[n + 1];

cout << "Введите строку" << endl;

cin >> a; // считываем строку

zamena(a, n); // выполняем замену символов в соответствии с заданием

for (int i = 0; i < n; i++) { // выводим строку

cout << a[i];

}

cout << endl;

delete[] a; // освобождаем память

break;

}

case 3:{ // если пользователь выбрал выход, то прерываем программу

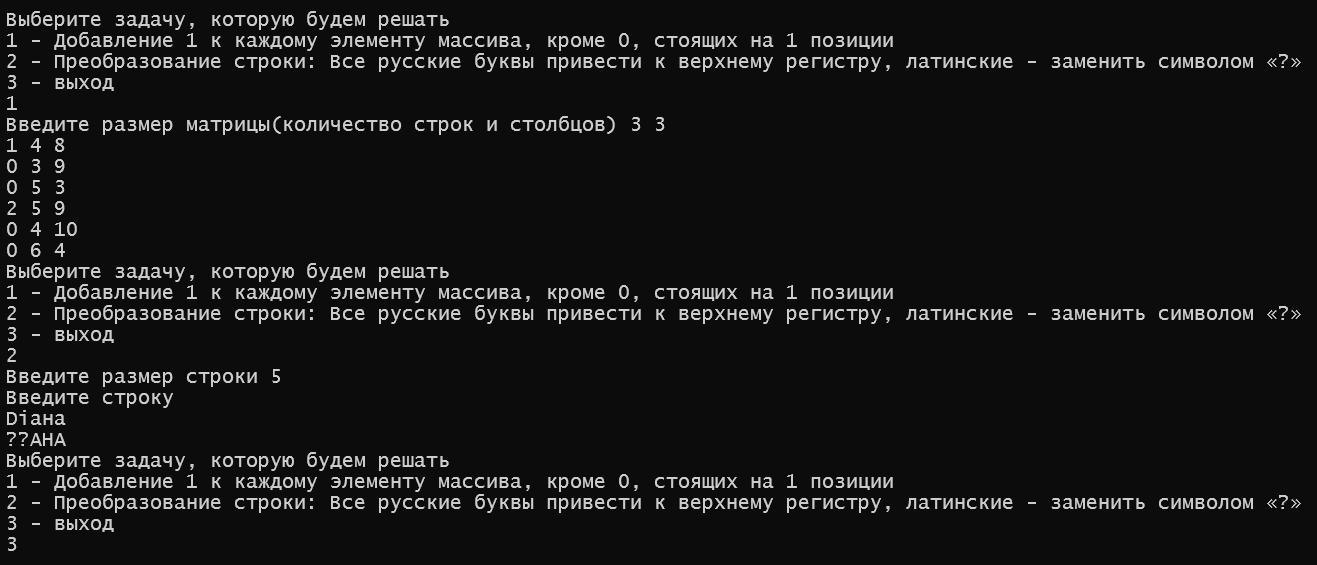
break;

}

}

} while (choice != 3);

}



|  |  |
| --- | --- |
| **3** | 1. Найти сумму элементов, лежащих ниже главной диагонали в целочисленном массиве **A**[**n**, **n**].  2. Написать программу, которая вводит несколько строк текста и символ «**m**» и использует функцию, чтобы определить суммарное число вхождений символа в текст. |

#include <iostream>

#include <iomanip>

#include <Windows.h>

#include <stdio.h>

#include <conio.h>

using namespace std;

int sumn(int \*\*a, int n, int &sum) { // функция для подсчёта суммы элементов матрицы, которые стоят ниже главной диагонали

for (int i = 0; i < n; i++) {

for (int j = 0; j < i; j++) {

sum += a[i][j];

}

}

return sum;

}

int vvod(char \*\*a, int n, char b, int &kol) { // функция ввода строк и подсчёта количества вхождений введённого символа

for (int i = 0; i < n; i++) {

int x;

cout << "Введите длину строки ";

cin >> x;

a[i] = new char[x];

cin >> a[i];

for (int j = 0; j < x; j++) {

if (a[i][j] == b) {

kol += 1;

}

}

}

return kol;

}

int main() { // функция main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do{

cout << "Выберите задачу, которую будем решать" << endl;

cout << "1 - Нахождение суммы элементов, которые стоят ниже главной диагонали " << endl;

cout << "2 - Подсчёт вхождений символа, введённого с клавиатуры, в введенный построчно текст " << endl;

cout << "3 - выход" << endl;

cin >> choice; // считываем выбор пользователя

switch (choice) { // создаём оператор выбора между задачами и опцию выхода

case 1: {

int n, sum = 0;

cout << "Введите размер матрицы ";

cin >> n; // считываем размер матрицы

int \*\*a = new int\*[n];

for (int i = 0; i < n; i++) { // считываем матрицу

a[i] = new int[n];

for (int j = 0; j < n; j++) {

cin >> a[i][j];

}

}

sum = sumn(a, n, sum); // выполняем функцию sumn

cout << "Сумма элементов, которые стоят ниже главной диагонали = " << sum << endl;

delete[] a; // освобождаем память

break;

}

case 2: {

char \*\*a; // создаём указатель на массив массивов символов

char b;

cout << "Введите количество строк ";

int n, kol = 0;

cin >> n; // считываем количество строк

a = new char\*[n + 1];

cout << "Введите символ ";

cin >> b; // считываем символ

vvod(a, n, b, kol); // вызываем функцию для ввода строк и подсчёта количества вхождений символа в текст

cout << "Количество вхождение символа " << b << " равно " << kol << endl;

delete[] a; // освобождаем память

break;

}

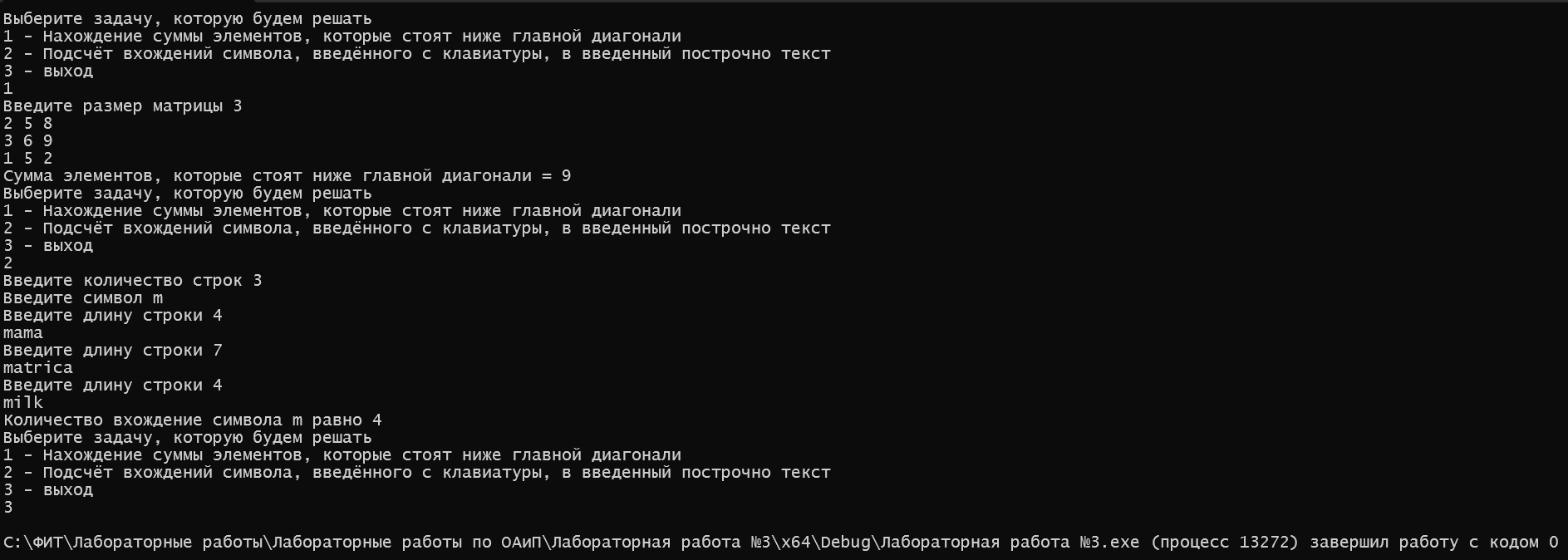
case 3:{ // если пользователь выбрал выход, то прерываем программу

break;

}

}

} while (choice != 3);

}

|  |  |
| --- | --- |
| **5** | 1. Дан двумерный массив.  В каждой строке выбрать минимальный элемент и среди них максимальный.  Напечатать номер строки, в которой расположен этот элемент.  2. Написать программу, которая вводит несколько строк текста и символ поиска и определяет суммарное число вхождений символа в текст. |

#include <iostream>

#include <iomanip>

#include <Windows.h>

#include <stdio.h>

#include <conio.h>

using namespace std;

int \*poisk\_min(int \*\*a, int n, int m, int \*b) { // функция для нахождения минимумов всех строк

int min;

for (int i = 0; i < n; i++) {

min = a[i][0];

for (int j = 0; j < m; j++) {

if (a[i][j] < min) {

min = a[i][j];

}

}

b[i] = min;

}

return b;

}

int poisk\_max(int\* b, int n, int &max, int &pos) { // функция для нахождения максимума среди минимумов

max = b[0];

pos = 0;

for (int i = 0; i < n; i++) {

if (b[i] > max) {

max = b[i];

pos = i;

}

}

return max;

}

int vvod(char \*\*a, int n, char b, int &kol) { // функция ввода строк и подсчёта количества вхождений введённого символа

for (int i = 0; i < n; i++) {

int x;

cout << "Введите длину строки ";

cin >> x;

a[i] = new char[x];

cin >> a[i];

for (int j = 0; j < x; j++) {

if (a[i][j] == b) {

kol += 1;

}

}

}

return kol;

}

int main() { // функция main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do{

cout << "Выберите задачу, которую будем решать" << endl;

cout << "1 - Нахождение максимального элемента среди минимальных из каждой строки " << endl;

cout << "2 - Подсчёт вхождений символа, введённого с клавиатуры, в введенный построчно текст " << endl;

cout << "3 - выход" << endl;

cin >> choice; // считываем выбор пользователя

switch (choice) { // создаём оператор выбора между задачами и опцию выхода

case 1: {

int n, m, max, pos = 0;

cout << "Введите размер матрицы ";

cin >> n >> m; // считываем размер матрицы

int \*\*a = new int\*[n];

for (int i = 0; i < n; i++) { // считываем матрицу

a[i] = new int[m];

for (int j = 0; j < m; j++) {

cin >> a[i][j];

}

}

int\* b = new int[n]; // создаём массив для минимумов

poisk\_min(a, n, m, b); // вызываем функцию для поиска минимумов из каждой строки

poisk\_max(b, n, max, pos); // вызываем функцию поиска максимума из минимумов

cout << "Максимальный элемент среди минимальных = " << max << endl;

cout << "Его первое вхождение находится в строке с индексом " << pos << endl;

delete[] a; // освобождаем память

break;

}

case 2: {

char \*\*a; // создаём указатель на массив массивов символов

char b;

cout << "Введите количество строк ";

int n, kol = 0;

cin >> n; // считываем количество строк

a = new char\*[n + 1];

cout << "Введите символ ";

cin >> b; // считываем символ

vvod(a, n, b, kol); // вызываем функцию для ввода строк и подсчёта количества вхождений символа в текст

cout << "Количество вхождение символа " << b << " равно " << kol << endl;

delete[] a; // освобождаем память

break;

}

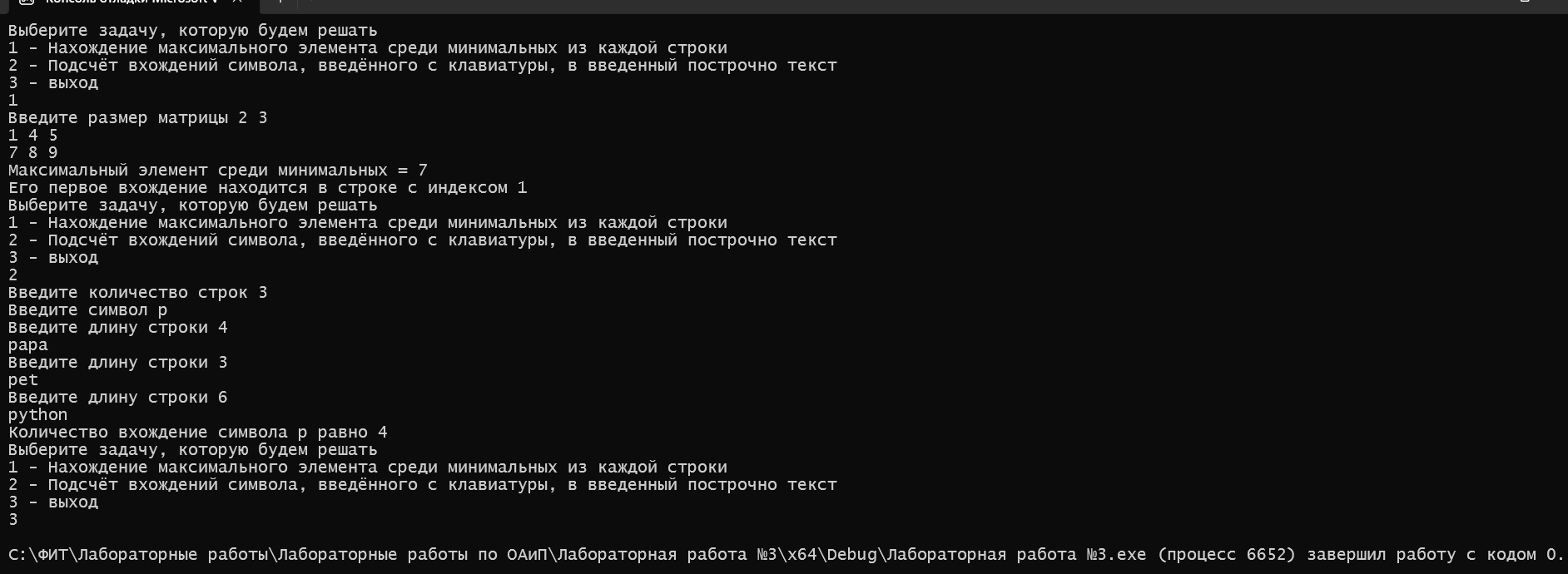
case 3:{ // если пользователь выбрал выход, то прерываем программу

break;

}

}

} while (choice != 3);

}